Data Structures and Algorithms

Lab 1 – C/C++ Primer

Question 1:

a) Compile and run four programs below to understand how reference works in C++;

Prog. 1:

```
#include <iostream>
using namespace std;
int main ()
{
  // declare reference variables
  int&    r = 10;
  cout << "Value of reference r : " << r  << endl;
  return 0;
}
```

Prog. 2:

```
#include <iostream>
using namespace std;
double value = 10;
double& passValue()
{
   return value;
}
int main ()
{
  // declare reference variables
  double interValue = 20.0;
  double&    r = interValue;
  cout << "Value of interValue reference : " << r  <<
endl;
  r = passValue();
  cout << "Value of value reference : " << r  <<
endl;
  return 0;
}
```

Prog. 3:

```
#include <iostream>
using namespace std;
double value = 10;
double& passValue()
{
   return value;
}
int main ()
{
  // declare reference variables
  double&    r ;
  r = passValue();
  cout << "Value of value reference : " << r  <<
endl;
  return 0;
}
```

Prog. 4;

```
#include <iostream>
using namespace std;
double& passValue()
{
   double value = 10;
   return value;
}
int main ()
{
  // declare reference variables
  double&    r = passValue();
  cout << "Value of value reference : " << r  <<
endl;
  return 0;
}
```

b) Given an enumerator below:

```
enum day
{
  Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday
};
```

Implement three operator s:

```
day *operator--(day &d);
day &operator++(day &d);
day const &operator++(day &d);
```

c) Write statement to describe the differences among three operators in question 2.b.

Question 2:

Consider the following class

```
class testClass
{
private:
  int x;
  int y;
  testClass *o;
public:
  int sum(); // return the sum of the private data members
  testClass (); // set each of x and y to 0.
  testClass (int a, int b); // set x and y to a and b, respectively.
  void getValue(int & a, int&b, testClass*& p);// set x to a and y to b.
  void createObj(int a, int b); // create an testClass Object and assign
                                //reference into pointer o.
};
```

a) Implement all methods as described in the comments

b) Create the destruction of test class in order to free all used variable.

Note: thinking in recursive delete. Is there any possible runtime error?

Question 3:

In a forest, the number of rabbits is represented by the following formula.

$$F(n) = F(n-1) + F(n-2) – F(n-3) ; n>0$$

where, F(n) is value at n-th month, F(1) = 2, F(2) = 4 and F(3) = 9;

Write a recursive function calculate amount of those rabbits at n-th month and print out how many is this recursive function called.

Question 4:

Give the quick sort implementation in c++ :

```
#include <iostream>
using namespace std;

const int INPUT_SIZE = 10;

// A simple print function
void print(int *input)
{
    for ( int i = 0; i < INPUT_SIZE;
```

```
// The quicksort recursive function
void quicksort(int* input, int p, int
r)
{
    if ( p < r )
    {
        int j = partition(input, p,
r);
        quicksort(input, p, j-1);
```

```
i++ )
        cout << input[i] << " ";
    cout << endl;
}

// The partition function
int partition(int* input, int p, int
r)
{
    int pivot = input[r];

    while ( p < r )
    {
        while ( input[p] < pivot )
            p++;

        while ( input[r] > pivot )
            r--;

        if ( input[p] == input[r] )
            p++;
        else if ( p < r )
        {
            int tmp = input[p];
            input[p] = input[r];
            input[r] = tmp;
        }
    }

    return r;
}
```

```
        quicksort(input, j+1, r);
    }
}

int main()
{
    int input[INPUT_SIZE] = {500, 700,
800, 100, 300, 200, 900, 400, 1000,
600};
    cout << "Input: ";
    print(input);
    quicksort(input, 0, 9);
    cout << "Output: ";
    print(input);
    return 0;
}
```

a) Adjust the code above to print out all quicksort function calls in order. Which is the caller of each recursive function execution.

b) Calculate the number of comparison and assignment has been operated by adding some statements.

Question 5:

Give a character set

{ A, B, C , D , E, F}

Write a function to print out all permutation. How many possible permutations are there?

Example:

A B C D E F

A C D B E F

….

F C B D E A